January 1996

IMAGE ACCESS FACILITY
INTEROPERABILITY ASSESSMENT REPORT

Central Imagery Office
United States Imagery System

8401 Old Courthouse Road
Vienna, Virginia 22182
U. S. A.

DRAFT

ABSTRACT

The Central Imagery Office (CIO) has the responsibility to establish the standards related to architectural elements within the United States Imagery System (USIS). Defining the interfaces between the software components of the USIS architecture makes up a part of this standards development task. The CIO, through work within the Image Access Working Group (IAWG), has adopted the use of the Object Management Group (OMG) Interface Definition Language (IDL) to specify these software interfaces. Directed research and development and technical support provided through the National Exploitation Laboratory (NEL) has resulted in the development of an IDL specification called the Image Access Facility (IAF). The IAF is a subset and first adopted facility of the overall Common Imagery Interoperability Facilities (CIIF). The CIIF and its facilities are provided to resolve interoperability problems and for data and service sharing across software boundaries. The IAF IDL is a specification that addresses interoperability needs between a virtual image archive and its clients. This report describes the successful testing of IAF sample implementations, including a detailed assessment of interoperability among these sample implementations.

EXECUTIVE SUMMARY

IMAGE ACCESS FACILITY GOALS

The Image Access Facility (IAF) is an evolving standard of the Central Imagery Office (CIO) for softcopy image access in the United States Imagery System (USIS). In an effort to identify a re-usable method for defining system interfaces for the USIS, the IAF was defined using a new computer language called Interface Definition Language (IDL), that permits system and technology independent definition of Interface Control Documents (ICDs). The IAF was successfully implemented and tested at the National Exploitation Laboratory (NEL) by three community contractors with the results documented in this report.

USIS STANDARDIZATION

The IAF is part of the USIS system engineering effort to achieve interoperability through the use of Common Imagery Interoperability Facilities (CIIF). A method was required to define system boundaries and incorporate legacy systems using the object-oriented design paradigm. The Object Management Group (OMG) Interface Definition Language (IDL) was selected as the optimal approach.

OBJECT MANAGEMENT GROUP (OMG) IDL

The OMG IDL is backed by the OMG consortium comprising over 500 industrial companies. The IDL is a notation for specifying software boundaries that are implementation independent, programming language independent, platform and operating system independent; it has been in use since its specification in August, 1991.

IAF IDL DEVELOPMENT PROCESS

Figure A-1 depicts relevant milestones in the development of the IAF IDL. The scope of the IAF was limited to image retrieval and update services for a virtual archive client interface. The scope did not include data and system management issues and a wide range of other USIS capabilities which may be addressed by other studies.

The Image Access Facility has four interfaces: Server, Product Request, Update Request, and Array Request. The purpose of the Server interface is to provide operations common to all other interfaces. The purpose of the Product Request is the retrieval of whole image products typically used by image product viewer and editor clients. The Update Request interface provides basic capabilities for adding image products to the archive. The interface is typically used by clients with authority and capability to update the archive. The Array Request interface provides pixel region retrieval images or other array data. The interface is used by image product viewers with optimized image product display, transmission and storage needs.

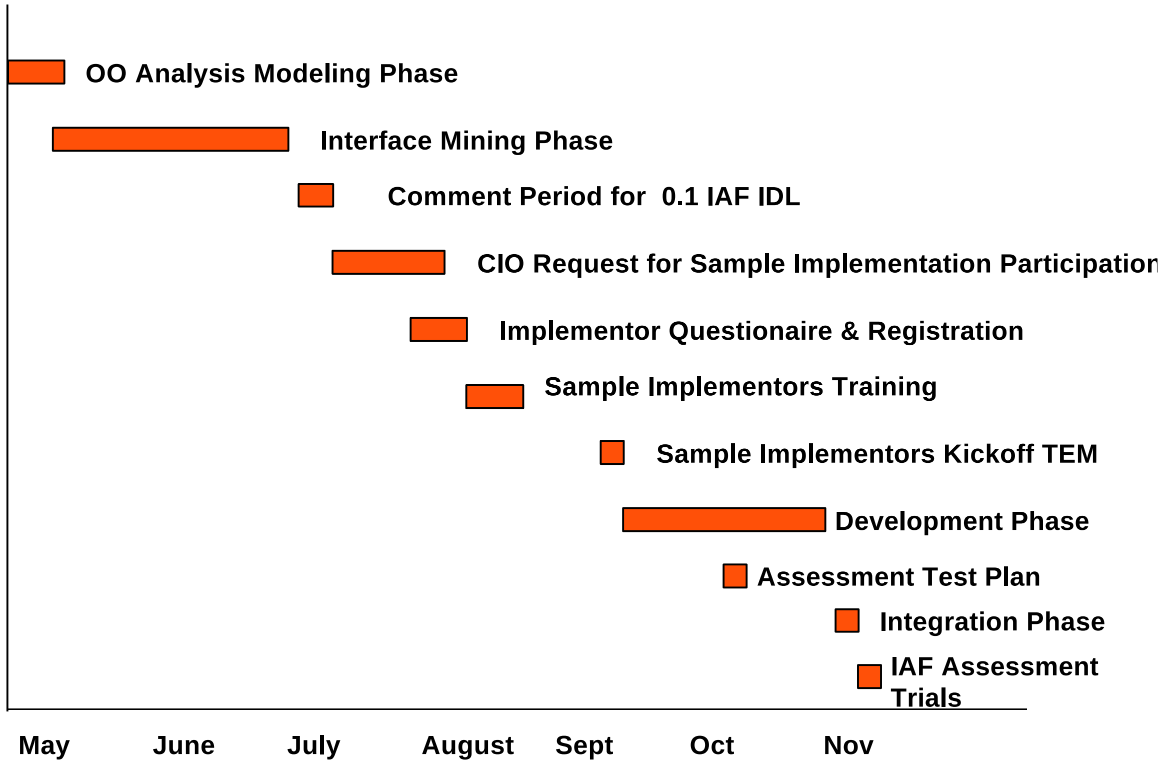

Figure A-1. IAF IDL Development Time Line

IAF IDL ASSESSMENT TRIALS

Three groups participated in the IAF assessment process. These groups included HITC Information Technology Corp (HITC), Autometric, and the contractor for the IMACTS system. The sample implementation participants began work in earnest after a technology exchange meeting on September 14, 1995. The version 0.7 "Silver" of the IAF IDL was selected and frozen for the duration of the sample implementation period. Seven weeks later, integration work for the interoperability assessment began (the week of November 6, 1995). The assessment was conducted and controlled through client invocation of server operations. “Successful” operation was defined as exception free for three invocations of three image products. The following list defines all of the tests that were conducted:

HITC Environment Operability
Autometric Environment Operability
IMACTS CONTRACTOR Environment Operability
HITC to Autometric Interoperability
Autometric to HITC Interoperability
HITC to IMACTS CONTRACTOR Interoperability
IMACTS CONTRACTOR to HITC Interoperability
Autometric to IMACTS CONTRACTOR Interoperability

IMACTS CONTRACTOR to Autometric Interoperability

The assessment activity involved efforts not only from the HITC, the IMACTS contractor and Autometric teams, but also from representatives of the NEL and MITRE. An initial Technical Exchange Meeting (TEM) was held with the integration assessment participants on October 24 at the NEL. Final testing plans were discussed and detailed technical information was exchanged. All participants arrived at the NEL on November 6 for porting and adjustments. Assessment trials began the afternoon of November 9. The work resumed the morning of November 13th. Final operability assessment of HITC IAF implementation was conducted at HITC on November 13th. All aspects of the the assessment including interoperability and operability invocations were successful for all participants.

CONCLUSIONS

The IAF IDL assessment trails were 100% successful with regard to the metrics taken. The concensus of the developers was that simplicity, interoperability and latitude in implementation were strong features of the IAF IDL. A general set of lessons learned during the IAF developement process was documented. A set of “level of effort” metrics was also recorded. The estimates of manpower required to support the IAF IDL sample implementations were reported for the development phase. Estimates of total staff weeks for full client/server implementation were categorized according to previous relevant experience. The results were as follows:

Experience		Staff weeks per
CORBA	C++	Operation Implementation
NO	NO	4
YES	NO	3
YES	YES	2

ACKNOWLEDGMENTS

The author wishes to acknowledge the contributions of the people supporting the IAF IDL effort. Special thanks to Dwight Brown, Steve Black, Ron Burns, Greg Cathell, Tim Daniel, Chris Deschenes, Ed Dorsie, Doug Everhart, Don Joder, Thomas Mowbray, Kathleen Perez-Lopez, Glenn Speckert, Doug Vandermade and Steve Weiss. Special thanks to the Autometric team including Ed Dorsie and Steve Weiss; the HITC team, including Greg Cathell, Kathy Saint and Scott Thomas; the IMACTS contractor team (who wish to remain anonymous).

TABLE OF CONTENTS

SECTION
PAGE

TABLE OF CONTENTS

SECTION ... PAGE

LIST OF Figures

LIST OF TABLES

SECTION 1

INTRODUCTION

The Central Imagery Office (CIO) has the responsibility to establish the standards related to architectural elements within the United States Imagery System (USIS). The CIO and the Image Access Working Group (IAWG) have initiated efforts to define the interfaces between the software components of the USIS architecture. The CIO has adopted the Object Management Group's (OMG's) Interface Definition Language (IDL) to specify these software interfaces. The CIO has proposed its first IDL specification called the Image Access Facility (IAF). The IAF is a subset and first adopted facility of the overall Common Imagery Interoperability Facilities (CIIF). The IAF IDL is a specification that addresses interoperability between a virtual image archive and its clients.

This report presents the results and lessons learned from sample implementations of the IAF IDL. It describes the technical aspects integration and assesses interoperability of the sample implementations. Section 2 of the document reviews the development schedule and
process that resulted in the proposed IAF. Section 3 describes process used to assess the interoperability and operability of the IAF IDL. Section 4 documents the lessons learned and guidance relevant to future versions of the IAF IDL. An Appendix is provided with photocopies of the actual data sheets used in the IAF IDL assessment trials. A bibliography and an acronym list are also provided.

SECTION 2

IAF IDL DEVELOPMENT PROCESS

2.1 IAF DEVELOPMENT GOALS

The IAF IDL is provided as an architectural specification language for legacy integration and interoperability between image archiving facilities within the USIS architecture. The short term goal is to demonstrate interoperability among multiple independently developed systems.

2.2 IAF IDL DEVELOPMENT

2.2.1 Common Interface Definition Process

The process used to define the IAF IDL interface involved both top-down and bottom-up analysis. The dual approach was used to assure a design that met USIS program requirements and could easily integrate legacy approaches.

The top-down process included requirements-driven object modeling and industry standards review. The process captured an object-oriented analysis model using the "CRC" methodology and utilized input from domain and requirements experts. Industry standards review included: analysis of the Z39.50 specification; review of the Harvest system; and pertinent OMG CORBAServices.

The bottom-up process included analysis and generalization of existing image archive software technology. The bottom-up survey included reviews of the Image Product Archive (IPA), System III, IMACTS/AIMS and other relevant technologies. The results of these studies contributed to the IAF specification.

The chart below (Figure 2-1) provides relevant milestones and schedule for activities of the IAF development process.

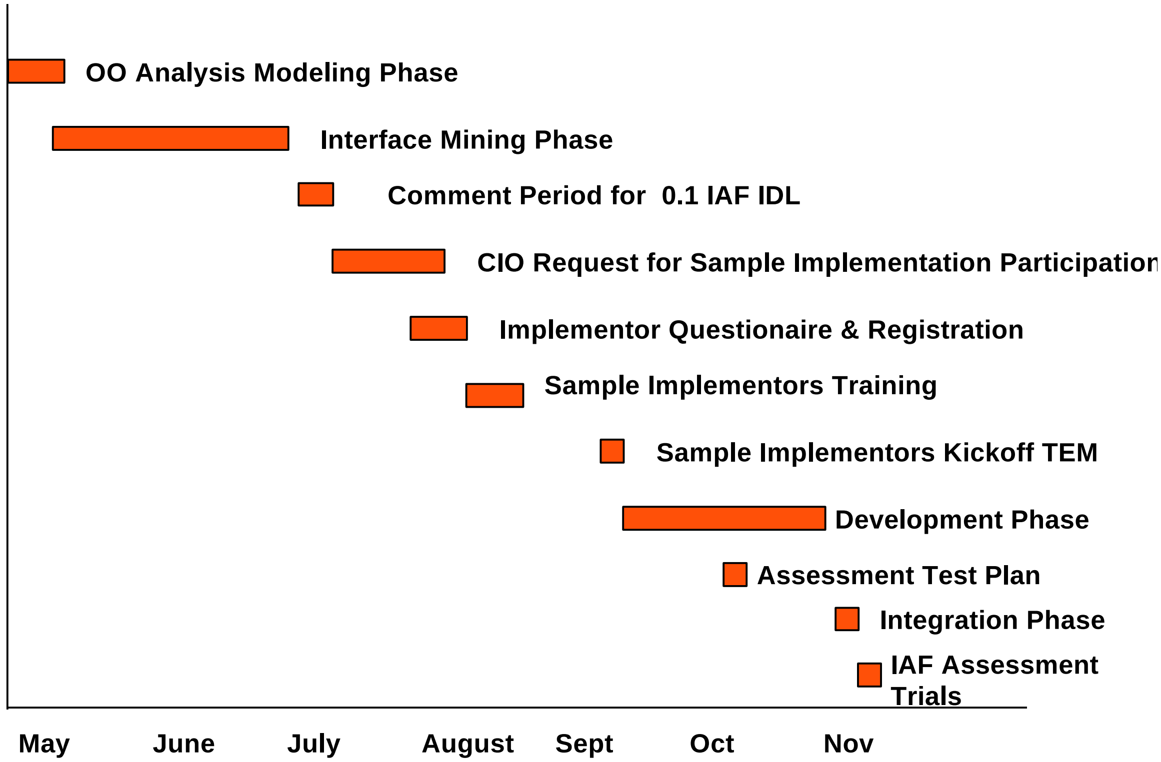

Figure 2-1 IAF IDL Development Time Line

2.2.2 IAF Interface Summary

The scope of the IAF was limited to image retrieval and update services for a virtual archive client interface. The scope did not include data and system management issues and a wide range of other USIS capabilities which may be addressed by other studies.

The IAF has four interfaces: Server, Product Request, Update Request, and Array Request.

The Server interface provides operations common to all of the other interfaces in the IAF. The interface includes the open and close operations. The open operation must be invoked in order to establish a login connection between client and server. The close operation is utilized to close the client server connection.

The Product Request interface provides for the retrieval of whole image products. The interface includes the disseminate, complete and cancel operations. The disseminate operation is used to request the transfer of an image product. The complete operation enables clients to check the status of an image product transfer request initiated through the disseminate operation. The cancel operation enables clients to abort outstanding requests initiated by the disseminate operation. The interface is typically used by image product viewer and editor clients.

The Update Request interface provides basic capabilities for adding image products to the archive. The interface comprises create, delete and copy operations. The create operation stores a new product in the image archive. The delete operation removes a product from the archive. The copy operation is used to duplicate an image product either locally or in a different archive. The interface is typically used only by clients with authority and capability to update the archive.

The Array Request interface provides pixel region retrieval images or other array data. The interface includes the open_array, close_array,get_region, cache_region and set_region operations. The open_array operation is provided to initiate access to an array object and to establish the kind of access desired by the client. The close_array operation is used to signal the deallocation of array resources when client utilization of an array is complete. The get_region operation is used to retrieve pixel data from an image in an archive. The cache_region operation provides a hint to the archive as to the next region that is likely to be retrieved by the client. The set_region operation modifies the pixels in an image product. The interface is used by image product viewers with optimized image product display, transmission and storage needs.

Image products referred to in the IAF specification comprise a wide range of image types and imagery derived documents. Forms may include: images, text, graphics, audio, video, and multimedia. The National Imagery Transmission Format (NITF) is a file format standard published by the Department of Defense (DoD). NITF can be described as an imagery compound document specification since it enables the storage of imagery and many image-related products in a single file. Support for NITF was required for participation in the interoperability portion of the IAF IDL assessment.

SECTION 3

IAF IDL ASSESSMENT

3.1 PURPOSE OF IAF IMPLEMENTATION ASSESSMENT

The sample implementation assessment activity was designed to evaluate the IAF IDL as an architectural specification language for legacy integration and interoperability. The assessment also verifies and validates the functionality and operability of the IAF.

The lessons learned from the sample implementations will be incorporated into the IAF IDL specification to improve its quality and enhance its operational benefits. A subsequent final version of the specification will published by the CIO.

The exercise of implementing and interoperating with IDL is expected to help broaden the skill levels and increase the knowledge base of the imagery community. These improvements in turn will help reduce risk and encourage a wider range of input and review for subsequent CIIF facility definition efforts.

3.2 IAF IMPLEMENTOR REGISTRATION QUESTIONNAIRE

An IAF Implementor registration questionnaire was distributed to all interested groups participating in the IAWG. The purpose of the questionnaire was to gather relevant information required to determine level of understanding and hardware and software availability. The questionnaire also indicated a minimum hardware and software configuration required. The configuration was used to specify the platform for the interoperability testbed. Information collected in the questionnaire included the following:

- Program Office site information
- Previous OMG development experience
- Development plans
- Image Access components of interest
- Explanation of plans for IDL Sample Implementation.
- Previous language experience
- Legacy migration plans
- Expected use of database technology

3.2.1 Recommended Hardware

Participation in the Sample Implementation program required the designation of a minimum of two machines. The minimum hardware requirements for the program were:

- CPU: SPARCstation 2, SPARCstation 10, or SPARCstation 20
- RAM: 24 Mbytes
- Disk: 100 Mbytes free space

3.2.2 Recommended Software

The minimum software requirements for the program were:

Sun SparcWorks 3.0.2 C++ Compiler
Iona ORBIX 1.3
DISCUS Technology Transfer Package (TTP) Release 1.2

3.3 IAF IDL SAMPLE IMPLEMENTOR TRAINING

3.3.1 Classroom Work

A one week training seminar was provided for all IAWG members interested in participating in the IAF IDL sample implementation program. The training was held at the MITRE Hayes facility the week of July 7 through 11. The training established a base skill level for implementation participants and relevant contributors. The course topics included:

- Tutorial Overview, Capability Model, Architecture Process
- Introduction to Software Architecture
- Evolution of Distributed Computing Mechanisms
- Migration to Distributed Objects
- OMG Overview
- The CORBA Specification
- IDL Tutorial
- DISCUS in Perspective
- Case Study: Architecture
- Overview of an ORB: ObjectBroker
- Case Study: Framework's C Binding
- Case Study: Framework Services
- Object Wrapping
- Case Study: Lessons Learned
- CORBA 2.0

- ORB Product Survey
- Case Study: Framework C++ Binding
- CORBA Migration Strategies
- OMG Object Services
- Managing Change
- Architecture Design & Coordination Options
- Profiling Abstraction Layers, & Development Frameworks
- Framework Design Process
- CORBA based Design Patterns
- OpenDoc

3.3.2 Laboratory Work

The lab work (i.e., hands-on training activity) included CORBA programming experience and compiling the IAF IDL. The final lab demonstrated the ProductRequest interface integrated with a World Wide Web (WWW) client interface. The lab work included the following:

- Lab 1: Orbix, IAF IDL compiling
- Lab 2: Client/Server Programming
- Lab 3: Conversion Service
- Lab 4: Conversion Using Trader
- Lab 5: Object Wrapper Gateway
- Lab 6: Wrapping an RPC Application
- Lab 7: IAF ProductRequest Interface Implementation with World Wide Web Interface

3.4 IDENTIFICATION OF IAF IDL IMPLEMENTORS

The participants in IAF Implementor Training were narrowed to those able to commit the required personnel and resources for the duration of the assessment. Three groups were able to meet the required commitments: HITC Information Technology Corp. (HITC); the IMACTS contractor; and Autometric.

3.5 IDL SPECIFICATION AGREEMENT

The IAF specification developed for the assessment was written in OMG IDL. The 0.7 "Silver" version of the specification was frozen for the duration of the assessment period. A Request for Change (RFC) process was used to effect group-wide acknowledgment and acceptance of changes.

3.6 SAMPLE IMPLEMENTOR KICKOFF TEM

A kickoff technical exchange meeting (TEM) was held on September 14, 1995 at the MITRE Hayes facility to coordinate and resolve logistics for a successful development the IAF IDL sample implementation. The meeting included a definition of the goals of the development phase and a general shared scenario for interoperability. Representatives of the groups participating provided a presentation of their intended approach for the development phase. A presentation was also provided by the NEL describing the hardware and software resources available in the NEL's Exploitation Systems Testbed.

3.7 IAF IDL SAMPLE IMPLEMENTATION DEVELOPMENT PHASE

The sample implementation participants began work in earnest the TEM on September 14, 1995. Integration work for the interoperability assessment began the week of November 6, 1995.

3.7.1 Problems Encountered During Development Phase

Questions regarding all aspects of development were recorded. Problems experienced during the period included the following:

- Assumptions about freeing pixel data returned in ArrayRequest::get_region
- Freeing data allocated for user defined and system exceptions
- Required support of inherited operations
- CORBA::string_to_object() return value CORBA::Object narrow() required
- Set/get of reference_data of CORBA::Object
- Confirmation of version level of IDL for development
- BOA::create() of objects
- System calls within Orbix code
- Client confirmation of existing connection in Server::open()
- Use of C libraries in C++ code
- UpdateRequest::delete() conflicts with C++ reserved word

3.7.2 Staffing Estimates During Development Phase

Estimates of staffing required to support the IAF IDL sample implementation were reported for the development phase. Estimates of total staff weeks for full client/server implementation were categorized according to previous relevant experience. The results were as follows:

Experience		Staff weeks per
CORBA	C++	Operation Implementation
NO	NO	4
YES	NO	3
YES	YES	2

3.8 IAF IDL ASSESSMENT TEST PLAN

A test plan for the IAF IDL assessment was published on September 28, 1995. The draft test plan provided a framework and rationale for the verification and validation of the IAF IDL.

3.8.1 Objectives

The objectives of the IAF Interoperability Assessment were:

- To provide a proof of concept evaluation the IAF IDL as an architectural specification language for the USIS project.
- To provide proof that the IAF is the proper interface to provide the functionality required for CIO image archives.
- To provide proof that the IAF IDL provides legacy integration and interoperability.

3.8.2 Assessment Criteria

In order to verify the IAF IDL as a mechanism for interoperability across multiple image archive approaches, a definition for successful interoperation was required. In addition the verification of IAF IDL functionality and operability required a definition of successful operation.

3.8.2.1 Definition of Interoperability

For the purposes of the IAF assessment, interoperability is: the ability to successfully communicate in terms of syntax and semantics across heterogeneous platforms and implementation approaches.

3.8.2.2 Definition of Operability

For the purposes of the IAF assessment, operability is: successful, exception-free execution of the operations of an interface within a single development approach environment.

3.8.3 Logistics

3.8.3.1 Location

The location of the assessment testbed was agreed to be the NEL facility.

3.8.3.2 Time Frame

Integration work was originally set to commence the week of October 16, 1995. The date was subsequently moved to November 6, 1995. The assessment trials were originally set be conducted October 26 and 27, 1995. The date was subsequently moved to November 9, 1995.

3.8.3.3 Clearances

The assessment was conducted at the SECRET SI-TK level. Clearances and names of participants were provided to NEL security point of contact.

3.8.3.4 Expectations and Assumptions

The participants were expected to provide source code to simplify integration at the testbed. The participants were instructed to not bring compilers, debuggers, or software libraries to the testbed. All hardware and software needs outside of the default configuration were to be cleared through the NEL on-site point of contact. Image file formats were set to NITF. All participants were expected to follow the general interoperability scenario and support the interfaces required. Participants were expected to be present for all assessment trials.

3.8.3.5 Points of Contact

The NEL on-site point of contact was Steve Black. The off-site sample implementation lead was Tom Herron.

3.8.3.6 Assessment Trial Specifics

The test was controlled through client invocation of server operations. Successful operation was defined as three exception free invocations for three different image products.

For example, successful operation of the ProductRequest disseminate operation was measured as follows:

Product A		
	$\frac{\text{2 exception free invocations}}{\text{3 attempted invocations}}$	"Success"
Product B		
	$\frac{\text{1 exception free invocation}}{\text{3 attempted invocations}}$	"Failure"
Product C		
	$\frac{\text{3 exception free invocations}}{\text{3 attempted invocations}}$	"Success"

A trial measured success as a simple “best of three”. The case above would be considered successful. The trials were conducted by participants, recorded by

MITRE and witnessed by the NEL. The following list defines all of the trials that were conducted:

- HITC Environment Operability
- Autometric Environment Operability
- IMACTS Contractor Environment Operability
- HITC to Autometric Interoperability
- Autometric to HITC Interoperability
- HITC to IMACTS CONTRACTOR Interoperability
- IMACTS CONTRACTOR to HITC Interoperability
- Autometric to IMACTS CONTRACTOR Interoperability
- IMACTS CONTRACTOR to Autometric Interoperability

3.9 IAF IDL INTEROPERABILITY ASSESSMENT PHASE

IAF IDL assessment was conducted at the NEL facility between November 6 and 13, 1995.

3.9.1 IAF IDL Testbed Configuration

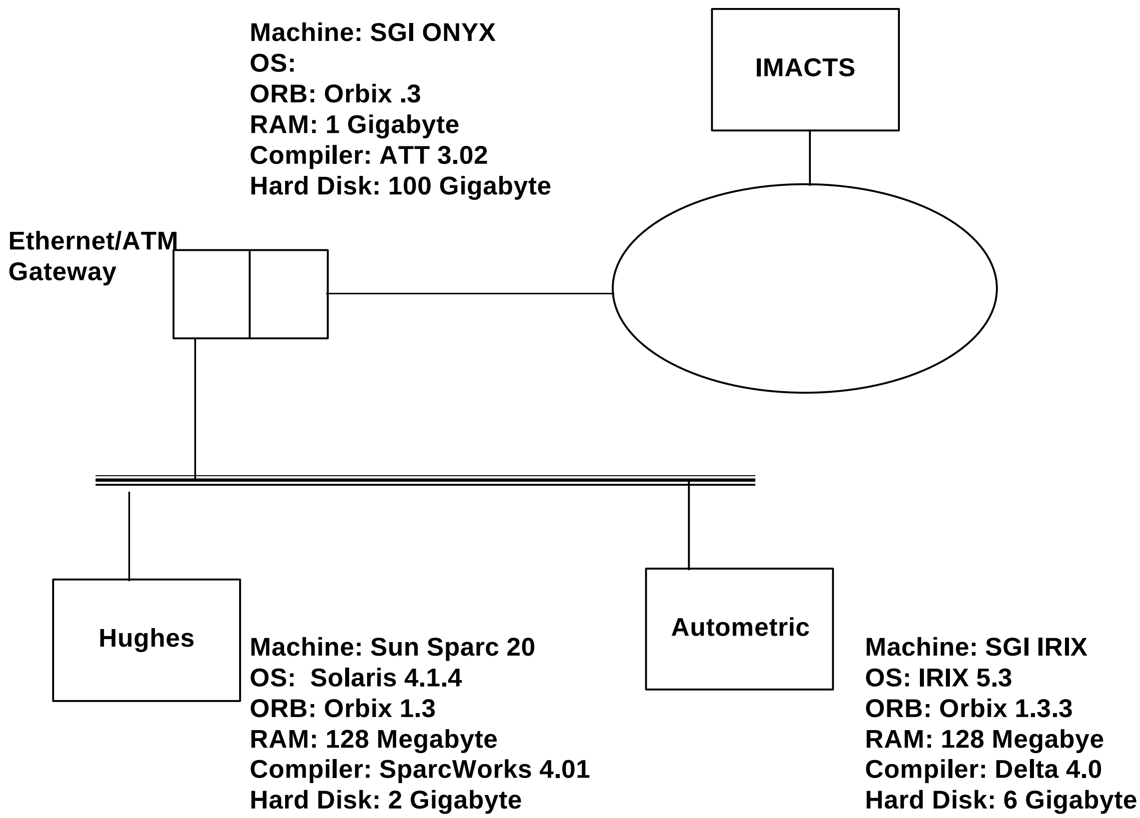

Figure 3-1 IAF Assessment Testbed

3.9.2 Integration Work

Integration work at the NEL was conducted between November 6 and 8, 1995. The integration work consisted of code finalization, testing and porting. Minor interoperability tests were conducted during the period.

3.9.3 IAF IDL Assessment Trial Results

The assessment trials were conducted at NEL and controlled through client invocation of server operations. The results are presented in Table 3-1.

<table>
<tr><td>Server::</td><td>Hughes Server</td><td>Autometric Server</td><td>IMACTS Server</td></tr>
<tr><td>open()</td><td>P1 P2 P3</td><td>P1 P2 P3</td><td>P1 P2 P3</td></tr>
<tr><td></td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T5</td></tr>
<tr><td>Hughes client</td><td>P P | P P | P P</td><td>P P | P P | P P</td><td>P P | P P | P P</td></tr>
<tr><td>Auto client</td><td>P P | P P | P P</td><td>P P | P P | P P</td><td>P P | P P | P P</td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>Server::</td><td>Hughes Server</td><td>Autometric Server</td><td>IMACTS Server</td></tr>
<tr><td>close()</td><td>P1 P2 P3</td><td>P1 P2 P3</td><td>P1 P2 P3</td></tr>
<tr><td></td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td></tr>
<tr><td>Hughes client</td><td>P P | P P | P P</td><td>P P | P P | P P</td><td>P P | P P | P P</td></tr>
<tr><td>Auto client</td><td>P P | P P | P P</td><td>P P | P P | P P</td><td>P P | P P | P P</td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>ProductRequest::</td><td>Hughes Server</td><td>Autometric Server</td><td>IMACTS Server</td></tr>
<tr><td>disseminate()</td><td>P1 P2 P3</td><td>P1 P2 P3</td><td>P1 P2 P3</td></tr>
<tr><td></td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td></tr>
<tr><td>Hughes client</td><td>P P | P P | P F P</td><td>P P | P P | P P</td><td>P P | P P | P P</td></tr>
<tr><td>Auto client</td><td>P P | P P | P P</td><td>P P | F P P | P P</td><td>P P | P P | P P</td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>UpdateRequest::</td><td>Hughes Server</td><td>Autometric Server</td><td>IMACTS Server</td></tr>
<tr><td>create()</td><td>P1 P2 P3</td><td>P1 P2 P3</td><td>P1 P2 P3</td></tr>
<tr><td></td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td></tr>
<tr><td>Hughes client</td><td>P P | P P | P P</td><td>P P | P P | P P</td><td>P P | P P | P P</td></tr>
<tr><td>Auto client</td><td>P P | P P | P P</td><td>P P | P P | P P</td><td>P P | P P | P P</td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>UpdateRequest::</td><td>Hughes Server</td><td>Autometric Server</td><td></td></tr>
<tr><td>copy()</td><td>P1 P2 P3</td><td>P1 P2 P3</td><td></td></tr>
<tr><td></td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td></td></tr>
<tr><td>Hughes client</td><td>P P | P P | P P</td><td>P P | P P | P P</td><td></td></tr>
<tr><td>Auto client</td><td></td><td>P P | P P | P P</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>UpdateRequest::</td><td>Hughes Server</td><td></td><td></td></tr>
<tr><td>delete()</td><td>P1 P2 P3</td><td></td><td></td></tr>
<tr><td></td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td></td><td></td></tr>
<tr><td>Hughes client</td><td>P P | P P | P P</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>ArrayRequest::</td><td>Hughes Server</td><td></td><td></td></tr>
<tr><td>get_region()</td><td>P1 P2 P3</td><td></td><td></td></tr>
<tr><td></td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td></td><td></td></tr>
<tr><td>Hughes client</td><td>P P | P P | P P</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>ArrayRequest::</td><td>Hughes Server</td><td></td><td></td></tr>
<tr><td>set_region()</td><td>P1 P2 P3</td><td></td><td></td></tr>
<tr><td></td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td></td><td></td></tr>
<tr><td>Hughes client</td><td>P P | P P | P P</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>ArrayRequest::</td><td>Hughes Server</td><td></td><td></td></tr>
<tr><td>cache_region()</td><td>P1 P2 P3</td><td></td><td></td></tr>
<tr><td></td><td>T1 T2 T3 | T1 T2 T3 | T1 T2 T3</td><td></td><td></td></tr>
<tr><td>Hughes client</td><td>P P | P P | P P</td><td></td><td></td></tr>
</table>

Key: P1 = Product #1; T1 = Test #1 etc.

TABLE 3-1 IAF IDL ASSESSMENT TRIAL RESULTS

3.9.3.1 Server::open Operation Results

The open operation was supported on the server side by all participants. Client support was provided by HITC and Autometric. Implementation of the open operation consisted of a check of user name and password.

Verification of operability was made through exceptions returned for invalid name and password. Verification of interoperability was made through system exception and screen print statements.

All measurements resulted in acceptable outcomes for the first two invocations for three invocations within each trial. The third invocation was not required.

3.9.3.2 Server::close Operation Results

The close operation was supported on the server side by all participants. Client support was provided by HITC and Autometric. Implementation of the close operation consisted of screen print statements.

Verification of operability was made through screen print statements. Verification of interoperability was made through system exception and screen print statements.

All measurements resulted in acceptable outcomes for the first two invocations for three invocations within each trial. The third invocation was not required.

3.9.3.3 ProductRequest::disseminate Operation Results

The disseminate operation was supported on the server side by all participants. Client support was provided by HITC and Autometric. All participants implemented the QUEUED disseminate operation. Autometric and IMACTS utilized scripted file transfer protocol (ftp) approaches while HITC utilized a freeware ftp product called "Expect".

Verification of operability was made through file system listings of file size and date. Visual verification was also provided by Autometric and HITC clients. Verification of interoperability was made through file system listings of file size and date, system exception and screen print statements.

All measurements resulted in acceptable outcomes for the first two invocations for three invocations within each trial. The third invocation was required for the Autometric client invocation of HITC server. The HITC ftp freeware package had an unexpected time-out setting that caused large files (> 8 megabytes) to fail. The time-out setting was adjusted and no additional problems were encountered.

3.9.3.4 ProductRequest::complete Operation Results

The complete operation was not supported by any of the participants. All participants implemented the QUEUED disseminate operation. The QUEUED disseminate operation is synchronous and does not return a transaction code required to use the complete operation. The complete operation was therefore left unimplemented by all participants.

3.9.3.5 ProductRequest::cancel Operation Results

The cancel operation was not supported by any of the participants. All participants implemented the QUEUED disseminate operation. The QUEUED disseminate operation is synchronous and does not return a transaction code as as required to use the cancel operation. The complete operation was therefore left unimplemented by all participants.

3.9.3.6 UpdateRequest::create Operation Results

The create operation was supported on the server side by all participants. Client support was provided by HITC and Autometric. Autometric and IMACTS utilized scripted ftp approaches while HITC utilized a freeware ftp product called "Expect".

Verification of operability was made through file system listing of file size and date. Visual verification was also provided by Autometric and HITC clients. Verification of interoperability was made through file system listing of file size and date, system exception and screen print statements.

All measurements resulted in acceptable outcomes for the first two invocations for three invocations within each trial. The third invocation was not required.

3.9.3.7 UpdateRequest::copy Operation Results

The copy operation was supported on the server side by all participants. Interoperable client support was provided by HITC. All implementations utilized the create operation on the target archive server.

Verification of operability was made through file system listings of file size and date. Visual verification was also provided by Autometric and HITC clients. Verification of interoperability was made through file system listings of file size and date, system exception and screen print statements.

All measurements resulted in acceptable outcomes for the first two invocations for three invocations within each trial. The third invocation was not required.

3.9.3.8 UpdateRequest::delete Operation Results

The delete operation was supported on the server and client side by HITC. Implementation consisted of file system removal of the target product. Additional catalog maintenance was also provided.

Verification of operability was made through file system listings of file size and date. Verification of interoperability was not required.

All measurements resulted in acceptable outcomes for the first two invocations for three invocations within each trial. The third invocation was not required.

3.9.3.9 ArrayRequest::open_array Operation Results

The open_array operation was not supported by any of the participants. The ArrayRequest interface was implemented by HITC but allocation of server resources was not required. The open_array operation was therefore left unimplemented.

3.9.3.10 ArrayRequest::close_array Operation Results

The close_array operation was not supported by any of the participants. The ArrayRequest interface was implemented by HITC but deallocation of server resources was not required. The open_array operation was therefore left unimplemented.

3.9.3.11 ArrayRequest::get_region Operation Results

The get_region operation was supported on the server and client side by HITC. Implementation consisted of pixel transfer of a subregion of a target array.

Verification of operability was made through average pixel density of the subregion and file system listing of file size and date. Visual verification was also provided by HITC clients. Verification of interoperability was not required.

All measurements resulted in acceptable outcomes for the first two invocations for three invocations within each trial. The third invocation was not required.

3.9.3.12 ArrayRequest::set_region Operation Results

The set_region operation was supported on the server and client side by HITC. Implementation consisted of pixel transfer of a subregion of a target array and the insertion of the sub_region into the original array.

Verification of operability was made through average pixel density of the subregion and file system listing of file size and date. Visual verification was also provided by HITC clients. Verification of interoperability was not required.

All measurements resulted in acceptable outcomes for the first two invocations for three invocations within each trial. The third invocation was not required.

3.9.3.13 ArrayRequest::cache_region Operation Results

The cache_region operation was supported on the server and client side by HITC. Implementation consisted of pixel cache of a subregion of a target array.

Verification of operability was made through average pixel density of the subregion. Verification of interoperability was not required.

All measurements resulted in acceptable outcomes for the first two invocations for three invocations within each trial. The third invocation was not required.

SECTION 4

LESSONS LEARNED AND GUIDANCE

4.1 IAF IDL IMPLEMENTOR QUESTIONNAIRE

A questionnaire was provided to each of the developers participating in the IAF IDL sample implementation. The questionnaire was provided to obtain feedback about technical and conceptual problems encountered developing with the IAF IDL.

Q1)
What was the most difficult problem encountered involving the IAF during the sample implementation development?

A1.a) My biggest problem with the IDL during development was finding an appropriate error return code. The IDL didn't support a full range of error codes, especially for create, copy and disseminate.

A1.b) Understanding the language mapping from IDL to C++ and properly manipulating IDL constructs (sequences in particular).

A1.c) How to handle the code for the Product and Array types. Initially misunderstanding the differences between the disseminate() and copy() operations. How the copy() operation works.

A1.d) The most difficult problem involving the IAF IDL specification during the sample implementation period was implementing the “opaque” product reference. Some view this as implementation specific, but it is domain specific. This opaque product reference resulted in the problem of how to pass image identifiers. There is no way of doing this in a clear consistent manner given the present IAF IDL specification (no markers).

Q2)
What was the most difficult problem encountered involving the ORB during the sample implementation development?

A2.a) My biggest problem with Orbix was understanding the concept of markers, manipulating the markers, and including application specific information in the markers.

A2.b) Persistence of objects and keeping the client/server connection active.

A2.c) How to handle persistent objects. IONA's concept of loaders to provide a dynamic method of handling persistent objects versus the static approach of handling persistent objects.

A2.d) Resolving the ambigous use of the “::” syntax used in implementation classes of the Orbix generated C++ was a particularly tough problem to solve. The “::” operator has a dual use, one involves inheritance, the other static members of classes. One has to be careful using this operator.

Q3)
What was the most difficult concept you struggled with involving the IAF IDL during the development period?

A3.a) The most difficult IAF IDL concept during development was determining the difference between copy and create. These two API's seem to be very similar.

A3.b) The fact that the Product interface was opaque. It took quite a while to grasp how to implement and use something with essentially no definition.

A3.c) The opaqueness of the Product and Array types. It is still unclear to me the use of '_i' appended to definitions.

A3.d) Conceptually, how the opaque product fits is not clear.

Q4)
What was the most difficult concept you struggled with involving the ORB/CORBA during the development period.

A4.a) The most difficult ORB/CORBA concept during development was the CORBA_Environment parameter. I wasn't quite sure how to set an error.

A4.b) The differences between the language interface created by the IDL compiler and our interface, and when to use each one for what functionality.

A4.c) How CORBA objects are handled, what are they, and how a persistent CORBA object is handled.

A4.d) Nothing comes to mind, conceptually CORBA is straight forward.

Q5)
What was the most difficult problem encountered involving the IAF IDL during the integration exercise?

A5.a) None.

A5.b) None.

A5.c) Inheritance of the BOAImpl was no put in the iaf.h file by the IDL compiler - this caused problem and we added other definition to our iaf.h file.

A5.d) The filename/image identifier state changes were difficult to track. The generation of a clock based product name was required for the IMACTS ingest to avoid an IMACTS exception due to possible product duplication attempts in IMACTS. This was driven from the scope and definition of the demonstration. A more general and robust solution is required before the IAF should be considered complete.

Q6)
What was the most difficult problem encountered involving the ORB/CORBA during the integration exercise?

A6.a) During integration, each server had to have a "pre-defined" interface marker. Using these markers became a little messy.

A6.b) ORB seemed to not invoke our server interfaces for our client. Not sure why.

A6.c) Memory allocation - we often had to allocate memory to CORBA or IA objects in order for our code to work properly - even in cases that shouldn't require allocation.

A6.d) The most difficult problem with the ORB was determining that the daemons could not communicate because of the environment. This problem arose because the IMACTS machine was not set up with the default Orbix configuration.

Q7)
What recommendations involving changes to the IAF IDL would you make given your experiences during the sample implementation

A7.a)

- open - none
- close - none
- disseminate - none
- create, copy - The two are very similar. One of them could be eliminated and a new single function could be defined. When a copy is performed, a create must occur for the new product.

A7.b)

- disseminate - make disseminate return useful info after queued service. Remove password for better security
- create - remove password for better security
- copy - remove password for better security
- delete - remove password for better security

A7.c)

- open, close, open_array, close_array - these are nice to have but not required by all implementations. To be 100% compliant, a statement should be made that all defined interfaces are mandatory for an implementation.
- disseminate - create copy - is there a way to hide the concept that to transfer a product you must make a file and use ftp? We are more comfortable with this approach than others are.
- delete - should be renamed since delete is a reserved word in C++.

A7.d) Provide an unabiguous way to refer to products by image identifiers. Generality in this case inhibits interoperability.

Q8)
What recommendations involving changing the sample implementation integration exercise would you make?

A8.a) I recommend providing the exact tests to the developers prior to the "real demo." Even though we successfully tested a HITC create to Autometric followed by a HITC copy from Autometric to HITC, I never considered this test during my development.

A8.b) More formal and defined testing. Supply all vendors with adequate and equal environments for final integration prior to test.

A8.c) Allow for a longer development time. We had a learning curve to first overcome. Definitely require more technical interchange meetings between implementors.

A8.d) Recommend better coordination among the developers. There is a risk in deferring implementation decisions to the last minute.

Q9)
What is your overall assessment of you experiences regarding the IAF IDL as an architectural language for interoperability and operability?

A9.a) Based on the code we developed, the IAF IDL seems to be a "so-so" architecture. I'm not overly impressed, nor do I disdain the IDL. We seem to have "exploited" the IDL beyond its initial design in order to compensate for heterogeneous databases. Did we really use the API's the way they were designed? Somehow the heterogeneous database issue seemed to get in the way of experiencing the IAF IDL.

A9.b) Strength in simplicity. It is a good general framework which allows interoperability and latitude in implementation. Some changes could improve it but it should be kept simple and general.

A9.c) Definitely a good start.

A9.d) Overall, the experience was useful. However before embracing CORBA or Orbix as the implementation medium for USIS, several issues need to be resolved. The number of details that had to be agreed to ad hoc by the involved parties seems to indicate that something more than a generally defined IDL is required to achieve interoperability.

1. A priori knowledge of the number of products to be passed around was required. This is not related to the catalog/discovery problem.
2. Markers were employed in an ad hoc basis to pass image/product ids. This is not a general solution. Use of markers and possibly even bind are Orbix specific.
3. What about other Orbix and communications between Orbs? Sure Orbix will become the standard but how can you convince the standards people of this?

APPENDIX: DATA SHEETS

Photocopies of the assessment trial data sheets are attached.

BIBLIOGRAPHY

1. *USIS Standards and Guidelines* -- *USIS Standards, Guidelines, and Conventions*, Section 4, Central Imagery Office, Vienna, VA, May 1995.

2. *National Imagery Transmission Format (NITF)*, Version 2.0, U.S. DoD MIL-STD 2500A.

3. *Interface Control Document for IPA 1.0*, GTE Government Systems Corporation, Document Number 1947004D, February, 1995.

4. *System III Application Programmers Interface*, CDR1 Update, Document Number 60770-6043-SX-00, July 1994.

5. *The Common Object Request Broker Architecture and Specification*, OMG Document Number 93.12.43, December, 1993.

6. *CARS Mission Intelligence Segment: Trusted Image Storage Manager* (Briefing), Loral Western Development Laboratories, San Jose, CA, June 1995.

7. Mowbray, T.J., and Zahavi, R., The Essential CORBA: System Integration Using Distributed Objects, John Wiley & Sons, New York, 1995.

8. *DISCUS Programmer's Reference Manual, Developer's Guide, and Release Notes*, The MITRE Corporation, McLean, VA, May 1995.

9. *ImNet with CORBA Extensions* (Briefing), The MITRE Corporation in cooperation with IMACTS, McLean, VA, June 1995.

10. *USIS Image Access Business Object Analysis* (Briefing), USIS Architecture Team, Vienna, VA, June 1995.

11. *USIS Image Access Standards Analysis* (Briefing), USIS Architecture Team, Vienna, VA, May 1995.

12. *Addendum: Summary of Migration Systems Analysis*, USIS Architecture Team, Vienna, VA, June 1995.

13. *United States Imagery System Master Glossary*, Version 2, Central Imagery Office, Vienna, VA, August 1994.

14. *Image Access Facility* Version 0.7, Central Imagery Office, Vienna, VA, September 1995.

ACRONYMS

AIMS	Array Information Management System
API	Application Program Interface
CARS	Collateral Archive Retrieval System
CAWS	Commercial Analyst Workstation
CDR	Critical Design Review
CIIF	Common Imagery Interoperability Facilities
CIO	Central Imagery Office
CORBA	Common Object Request Broker Architecture
COTS	Commercial off-the-shelf
DISCUS	Data Interchange and Synergistic Collateral Usage Study
DOD	Department of Defense
ESD	Exploitation Support Data
FTP	File Transfer Protocol
GIF	Graphics Interchange Format
GOTS	Government off-the-shelf
HTTP	Hypertext Transfer Protocol
IAF	Image Archive Facility (IDL module)
IAWG	Image Access Working Group
IDL	Interface Definition Language
IPA	Image Product Archive
ISO	International Standards Organization
NEL	National Exploitation Laboratory
NITF	National Imagery Transmission Format
NTB	NITF Technical Board
OMG	Object Management Group
ORB	Object Request Broker
RFC	Request for Change
SDE	Support Data Extension
SPIA	Standard Profile for Imagery Access (schema)
TEM	Technical Exchange Meeting
URL	Universal Resource Locator
USIS	United States Imagery System